IN THE SPECIFICATION

Please replace the paragraph beginning at line 22, page 12 with the following rewritten paragraph:

FIGURE 3 illustrates the principles of the invention. FIGURE 3(a) is, for the sake of clarity, the control flow of SOCKS Versions 4 and 5. The chief difference between the two versions is that Version 5 of SOCKS adds authentication [300] to Version 4 [305]. The mechanism of the present invention, illustrated in FIGURE 3(b), rather than performing the SOCKS protocol between client [310] and SOCKS server [320], which requires that each client be 'socksified', includes an agent [315], at the SOCKS server, playing the client's role on behalf of the client. Then, processes like authentication [325] takes place between this agent and a subscriber directory or database (e.g., [361]) where policies and generally all parameters specific to clients were previously stored. Hence, an initial connection request [345] from a client, captured by the transparent SOCKs manager [350], triggers an interrogation of the directory [346] which determines first what version of SOCKS client uses. Depending upon the result of this first interrogation client, the request is directed to SOCKS V5 agent [335] or SOCKS V4 agent [340]. When SOCKS V5 agent is selected, the directories [356] [330] and [361] are interrogated again to find what methods [355] are used by client and what kind of authentication parameters [360] are set. Then, (this is however the first step if SOCKS V4 was selected earlier), client agent [315] passes the request to the SOCKs server [320] which starts processing the connection request [370]. At this point, leading part of client's application data [375] may already have been obtained so that it can be thoroughly examined. This step, although optional, and which can be carried out at various levels of sophistication, opening the door to many possibilities that were not possible with standard SOCKS such as implementing a proxy cache for certain types of applications, for example HTTP previously described in FIGURE 1 in conjunction with transparent proxying. The leading part of the application data, which contains the headers of the protocols in use, can thus be examined and parsed to retrieve, through a further interrogation of the directory [376], all information necessary to process the application data and

application protocols used. For example, transparent SOCKS may thus determine what server (local, remote or none if request cannot be honored) is best suited, when several possibilities exist, to serve client requests and to keep using it consistently while the client session is on. Then, under normal circumstances, the SOCKS server establishes the connection with the application server and sends a circuit status to the SOCKS agent (which, however, has the freedom of resetting the connection with the client if something unexpected occurs on application server side). Finally, the SOCKS server establishes the data relay [380] between the application server and the client. This client is not aware that it is actually dealing with a transparent SOCKS and need not be configured.